**Flip Bits :-**

Easy Accuracy: 23.15% Submissions: 37K+ Points: 2

Given an array **A[]** consisting of **0’s** and **1’s**. A flip operation is one in which you turn **1** into **0** and a **0** into **1**. You have to do at most one “Flip” operation of any subarray. Formally, select a range **(l, r)**in the array **A[]**, such that (0 ≤ l ≤ r < n) holds and flip the elements in this range to get the maximum ones in the final array. You can possibly make zero operations to get the answer.

**Example 1:**

**Input:**

N = 5

A[] = {1, 0, 0, 1, 0}

**Output:**

4

**Explanation:**

We can perform a flip operation in the range [1,2]

After flip operation array is : [ *1* **1 1** *1 0* ]

Count of one after fliping is : 4

*[Note: the subarray marked in bold is the flipped subarray]*

**Example 2:**

**Input:**

N = 7

A[] = {1, 0, 0, 1, 0, 0, 1}

**Output:**

6

**Explanation:**

We can perform a flip operation in the range [1,5]

After flip operation array is : [ *1* **1 1 0 1 1** *1*]

Count of one after fliping is : 6

*[Note: the subarray marked in bold is the flipped subarray]*

**Your Task:**  
You don't need to read input or print anything. Your task is to complete the function **maxOnes()** which takes the array **A[]** and its size **N**as inputs and returns the maximum number of 1's you can have in the array after atmost one flip operation.

**Expected Time Complexity:** O(N)  
**Expected Auxiliary Space:** O(1)

**Constraints:**  
1 ≤ N ≤ 106  
0 ≤ A[i] ≤ 1

**Code :-**

//{ Driver Code Starts

#include<bits/stdc++.h>

using namespace std;

// } Driver Code Ends

class Solution{

public:

int maxOnes(int arr[], int n){

// kadane's algo = reward for converting 0->1, punish for 1->0

int sum=0, maxi=0, c1=0;

for(int i=0; i<n; i++){

if(arr[i] == 0)

sum += 1;

else{

sum -= 1;

++c1;

}

maxi = max(maxi, sum);

if(sum < 0)

sum = 0;

}

// cout<<"("<<maxi<<","<<c1<<")";

return maxi + c1;

}

};

//{ Driver Code Starts.

int main()

{

int t; cin>>t;

while(t--)

{

int n;

cin>>n;

int a[n+5];

for(int i=0;i<n;i++)

cin>>a[i];

Solution ob;

cout<< ob.maxOnes(a, n) <<endl;

}

return 0;

}

// } Driver Code Ends

**T.C :- O(N)**

**S.C :- O(1)**